Csernoch Mária – Bíró Piroska:

Spreadsheet misconceptions, spreadsheet errors

Absztrakt

A 2011/2012-es tanévben elindítottuk az Algoritmikus és Alkalmazói Készségek Tesztelése című projektünket, amelyben a Debreceni Egyetem Informatikai Karának elsőéves informatikus hallgatóit és gyakorló informatika tanárok informatikai ismereteit teszteltük. A cikkünkben a táblázatkezelési tesztek eredményeit ismertetjük, mivel korább már bizonyításra került, hogy a táblázatkezelői dokumentumok 95–60%-a hibás. A kutatásaink eredményei bizonyítják, hogy a táblázatkezelés során jelentkező hiányosságok nemcsak általános probléma, hanem az informatika szakos hallgatók és tanárok sem rendelkeznek megfelelő ismeretekkel. Elemzéseink során azt tapasztaltuk, hogy a népszerű felületkezelési módszer használata magyarázza a hibák nagy részét. Az eredmények ismertében a számítógépes problémák megoldására bevezettünk egy mély-szerkezetű metakognitív megközelítést. Az általunk bevezetett, a számítógépes tevékenységek megkülönböztetésére alkalmas kétféle metakognitív megközelítést – TAEW (trial-and-error wizard-based) és CAAD (computer-algorithmic- and debugging-based) – elhelyeztük Case és Gunstone metakognitív rendszerében. A cikk részletesen ismerteti a mély-szerkezetű megközelítés egy

lehetséges megvalósítását és megmutatja, hogy lényegesen hatékonyabb a felületkezelési megközelítéseknél.
Kulcsszavak: *táblázatkezelői dokumentumok hibái, képlet-létrehozási hibák, táblázatkezelés metakognitív megközelítése, tömbképletek*

Abstract

In the academic year of 2011/2012 Testing Algorithmic and Application Skills Project was launched to test the knowledge of Informatics of the first year students of the faculty of Informatics and teachers of Informatics in schools. This article focuses on spreadsheet problems, since previous studies have shown that 95-60% of the spreadsheet documents carry errors. Our tests clearly show that not just in general, but the students of Informatics have very limited spreadsheet knowledge, and so the teachers of Informatics. The article details the students' and teachers' results in the project and tries to find explanations for their underachievement. It was found that the widely used and commercialized surface approach methods for solving spreadsheet problems are the main source of the failure. It was also realized that there is a need for the categorization of the computer-related activities into the system of metacognitive approaches of Case and Gunstone. We introduced one surface (TAEW, trial-and-error wizard-based) and one deep (CAAD, computer-algorithmic- and debugging-based) approach method into the already accepted system. The article provides the details of our deep approach metacognitive method and its results, proving that

it is more effective than the previously accepted surface approach methods.[28]
Keywords: *spreadsheet errors, in-execution errors, metacognitive approaches to spreadsheet, array formulas*

Introduction

Spreadsheet programs appeared on the market around 30 years ago, and they are now among the most widely used programming systems (Scaffidi, Shaw, and Myers 2005). Originally, these programs were meant for domestic usage – a small program for calculating household expenses, handling personal data, and carrying out some minimal data retrieval (W1 2013; Sestoft 2010). However, time has proved that spreadsheet programs are more powerful and more widely used than was anticipated and communicated to the public. This contradiction has been inherent in the software from the very beginning, since the publishers have been boasting that these systems are easy-to-use programs, while at the same time continuously highlighting their large and increasing number of functions and powerful features.

[28] The research was supported by the TÁMOP-4.2.2.C-11/1/KONV-2012-0001 project. The project has been supported by the European Union, co-financed by the European Social Fund.
The research was supported partly by the Hungarian Scientific Research Fund under Grant No. OTKA K-105262.

Approaches to spreadsheets

Studies have shown that there is a high incidence of errors in spreadsheets; up to 90% in some cases (Abraham and Erwig 2009; Jorgensen 2013; Kadijevich 2009, 2013; Kwak 2013; Panko 2008; Panko and Aurigemma 2010; Powell, Baker and Lawson 2008, 2009a, 2009b; W2 2012; Teo and Tan 1999; Tort 2010; Tort, Blondel and Bruillard 2008). Studies have also been attempting to find an explanation for this failure. However, most of these researches have been carried out on completed and saved spreadsheets documents.

Metacognitive approaches to spreadsheets

It has also been proved and will be shown in this paper that a previously unknown metacognitive approach, which can be categorized as a surface approach, has emerged in the spreadsheet environment; the trial-and-error wizard-based approach (TAEW-based). With a paper-based testing method we would be able to prove that the TAEW-based approach is not sufficient to create correct spreadsheet formulas, and consequently to solve spreadsheet problems.

To solve spreadsheet problems algorithms have to be built and these algorithms have to be coded. The program codes have to be translated and after translation they have to be executed. To successfully complete spreadsheet problems another deep approach metacognitive method is needed (Case 2000; Case and Gunstone 2002, 2003; Case, Gunstone and Lewis 2001, Cox 2005; Csíkos 2006; Koriat and Levi-Sadot 2000); this method is the computer-algorithmic- and debugging-based (CAAD-based) approach.

Functions in Math and spreadsheets

The phenomenon of function used in Mathematics is adapted in spreadsheets. However, Microsoft, the producer of the most widely used spreadsheet system (Abraham and Erwig 2009), does not communicate the relationship between the two subjects, and consequently does not take advantage of the shared knowledge. This connectivity of functions in the two subjects should be emphasized in order to create correct formulas, especially multilevel, embedded formulas. Furthermore, using functions in real world problems would strengthen the students' background knowledge of these functions.

The impact of errors

Summarizing all these points, first we must state that spreadsheets are not programs for domestic use only. They are more powerful, and they can be used for serious amounts of data storage and information retrieval based on the stored data. This capability of spreadsheet programs has been clear from the very beginning and has been wildly used in businesses, small and large. This two-fold approach to the programs has resulted in spreadsheet documents –as was mentioned earlier – containing errors. However, error detection only starts in a serious form after the discovery that these errors are responsible for serious financial losses.

In-execution-error detection

Error detection is carried out mainly by automated error detection programs (Panko 2008; Panko and Aurigemma 2010). However, errors and problems are rooted deeper than completed formulas. Studies have shown the losses to companies caused by computer illiterate users, and the time they take to create documents (W7 2012).

In completed formulas we are not able to tell how many trials lead the user to the final product. In other words, how many clicks and how much time was needed for the user to arrive at a formula which is acceptable both to the compiler and the interpreter? Collecting data on how users create formulas and how users carry out debugging plays an important role in the process of teaching and creating spreadsheet documents.

There are at least two solutions to the task of testing the in-execution-errors. One of them is to create log files of the users' activities. The other solution is testing on paper. Both methods could provide data concerning the users' approaches to creating formulas but from a different point of view, and to test their different skills.

Methods

Paper-based testing

We have opted for the paper-based testing method to document the in-execution-errors, and to be able to follow the students' and teachers' knowledge of formulas without

any background support (Csernoch 2012; Csernoch and Bíró 2013a; 2013b;2013c).[29]

Simulation of the spreadsheet environment

In a paper-based test we have to simulate a spreadsheet environment with a sample table and tasks. To test the in-execution errors we first have to provide the table and the number of rows.

In this simulated environment the most commonly used tasks are those where spreadsheet formulas have to be created. The other task type is when a completed formula has to be evaluated and the answer should be a complete sentence in any natural language.

Figure 1 shows the first eight rows of the sample table in our test. Here a table of five columns with 216 rows is provided, in which the first row of the table contains the field names for the columns. The fields are the following: the name, the continent, the capital city, the size, and the population in thousands of the countries of the World.

[29] The research was supported by the TÁMOP-4.2.2.C-11/1/KONV-2012-0001 project. The project has been supported by the European Union, co-financed by the European Social Fund.
The research was supported partly by the Hungarian Scientific Research Fund under Grant No. OTKA K-105262.

Figure 1: The first eight rows of the sample table of the test

	A	B	C	D	E
1	Country	Continent	Capital	Area	Population (thousand)
2	Afghanistan	Asia	Kabul	647500	27756
3	Albania	Europe	Tirana	28748	3545
4	Algeria	Africa	Algiers	2381740	32278
5	American Samoa	Oceania	Pago Pago	199	69
6	Andorra	Europe	Andorra la Vella	468	68
7	Angola	Africa	Luanda	1246700	10593
8	Anguilla	Amerika	The Valley	102	12

Description of functions

Description and comparison of functions are tasks with which the users' background knowledge of spreadsheet functions can be tested. In spreadsheets these questions are vital because among the built-in functions several can be found with only seemingly minor differences, but severe restrictions.

Testing of spreadsheet knowledge

Sample

Our project was launched in the academic year of 2011/2012 and was repeated one year later (Csernoch and Bíró 2013a; 2013b;2013c). The first year students of the Faculty of Informatics of the EGYETEM NEVE were tested at the beginning of September, immediately after leaving high school and passing their maturation exams (360 and 370 students). The second phase of the project took place in the same semester after covering spreadsheets (around November), while the third phase one year later (the following November), when the students were in the second year of their university studies. Here, in this article

we will give details of the test in September and in November of the following year.
As well as testing our students, teachers of Informatics were also tested (134 teachers).

Tasks
Three different types of tasks were assigned to the students and teachers. The first two types were based on the sample table (Figure 1), while the third type was a comparison of functions.
In the first type four spreadsheet formulas had to be created to solve four problems:

- the capital city of the largest county (Task a),
- the population density of the countries (Task b),
- the number of African countries (Task c),
- the average population of those countries whose areas is smaller than H1 (Task d).

The second type of task was a completed formula, which provides the number of European countries whose initial letter is A (Task e);
{=SUM(IF(B2:B216="EUROPE";IF(LEFT(A2:A216)="A";1)))}
The third type of tasks was the comparison of the built-in HLOOKUP() and VLOOKUP() functions with the multilevel INDEX(MATCH()) function (Task f) (W4 2013).
We must note here that Tasks a) and f) required the same knowledge. The only difference between the two tasks was the way the question was framed.
Task b) did not require any knowledge of spreadsheet functions, but rather spreadsheet operators, some background knowledge from Math or Geography, some

data retrieved from the sample table, and the method(s) for finding the correct result for each of the countries.

Tasks c) and d) could be solved with the same method. Either by using one of the built-in functions – the *IF?() functions (W5 2013; W6 2013) or the database functions (W3 2013) – or creating multilevel conditional formulas (Csernoch and Balogh 2010; Walkenbach 2003; Walkenbach and Wilcox 2003; Wilcox and Walkenbach 2003). They both carry a relational operator, but the difference between the two tasks is that the relational operator in Task c) is =, while in Task d) it is <. There is a value in both tasks to be compared to the values in the table. The difference here is that while in Task c) a constant must be compared to the values in the table, in Task d) it is a variable. The problem that emerged here is that Microsoft with their built-in functions support mainly the equality and constants used in their own formulas. The other solution to the two similar tasks would be to use conditional single-result array formulas (CSRA), which are able to handle both the different conditions and the variables without any difficulties.

Considering all these points, we claim that the algorithmic- and/or TAEW-based surface approaches would be enough to solve the simplified Task c). However, to solve Task d), which requires more complicated input than Task c), a deep approach method (Case and Gunstone 2002, 2003), the DAAC-based method, is needed. We must note here, that the algorithmic-based and the computer-algorithmic- and debugging-based (CAAD-based) methods are different; the first is a surface approach, while the other a deep-approach.

Task e) is a CSRA formula to translate into a natural language. The students and teachers were asked to decode the formula and write down what its output would be.

Results

Tests in September

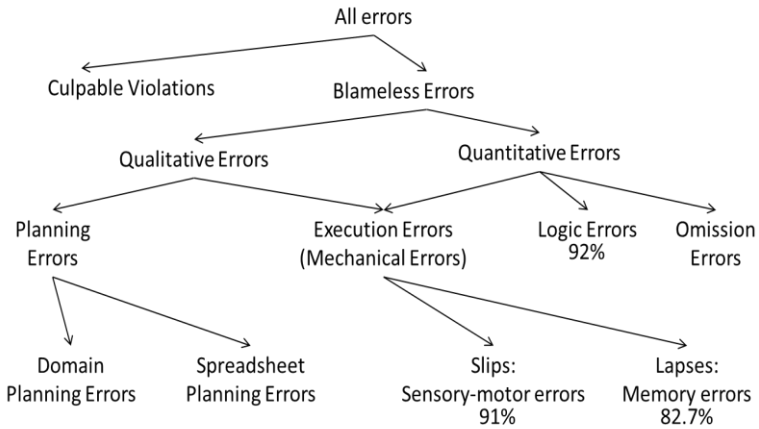
The percentage of the correct answers in the students' tests in September and the teachers' test are presented in Table 1 (Csernoch 2012; Csernoch and Bíró 2013a; 2013b;2013c). It is clear from the data that only a very low percentage of correct answers were given in the tasks. Even the teachers' results were extremely low, 19.95% on average for Tasks a) – e). Neither the students nor the teachers were able to answer correctly Task f), the description and comparison of the given functions.

There is one datum on the table which must be emphasized; the teachers' result of 11.94%. This is the percentage of those teachers who claimed that the formula of Task e) is incorrect and there is no solution to Task e). The formula was correct; the fact was simply that the teachers were not able to understand it.

Table 1: The percentage of correct answers in the students' tests in September and in the teachers' test. The bottom row shows the percentage of those teachers who claimed that the formula of Task e) was incorrect

	N	a)	b)	c)	d)	e)	f)
2011/2012	360	0.56%	1.11%	5,56%	0.83%	15.28 %	0.00%
2012/2013	370	0.00%	0.27%	8.64%	1,35%	24,87 %	0.00%
teachers	134	18.66 %	16.42 %	38.81 %	4.48%	31.34 %	0.00%
		19.95%					
						11.94 %	

Figure 2: The modified version of Panko and Aurigemma's (2010) taxonomy of development of testing error types



After evaluating the correct answers we focused on the different types of in-execution-errors, trying to find explanations for the low results. To categorize the errors we used Panko and Aurigemma's taxonomy of errors (Panko and Aurigemma 2010). However, we must mention here that they were only analyzing completed formulas and spreadsheets so their taxonomy does not fit perfectly with the in-execution-errors; the tendency, however, is clear. For further analysis of in-execution errors this taxonomy needs to be revisited.

Figure 2 clearly shows the high percentage of errors appearing in the formulas.

To solve Task a) three functions have to be used to build a three-level function (W4 2013). The innermost function is the MAX() (1 argument); outside of this the MATCH() function (3 arguments), and outside of the MATCH() function the INDEX() (2 or 3 arguments) function should be used. The data in

Table 2 clearly shows that both the students and the teachers had problems remembering the name of the functions. The best result was found for the one-argument MAX() function. However both MATCH() and INDEX() are hardly known at all by the students, and only around half of the teachers know the name of these functions. The teachers' results in remembering these functions are rather surprising because they are basic syllabus requirements of the ECDL and the maturation exams. The other discouraging data is that a relatively high number of teachers were able to remember the name of the functions but were not able to compose a three-level formula (compare results in Table 1 and

Table 2).

Table 2: The name(s) of the functions remembered by the students and the teachers when solving Task a)

	INDEX()	MATCH()	MAX()
2011/2012	17%	18%	46%
2012/2013	14%	13%	45%
teachers	49%	48%	63%

By checking whether the names of the functions in Tasks c) and d) were known we separated the many-argument built-in functions – *IF?() and database functions – from the CSRA formulas, in which simple functions had to be used to compose two-level functions.

When solving Task c) both the students and the teachers preferred to use the built-in *IF?() functions. However, more students tried to solve the problem with the multilevel function than was the case with the teachers. When solving Task d) the students ignored the built-in functions and tried to use the multilevel formula. The number of those who tried the built-in functions dropped markedly from Task c) to d). However, there are only minor differences between the uses of the simple functions which were built into multilevel formulas in the two tasks.

Far fewer teachers were able to remember the built-in function for Task d) than for Task c). The number of those who thought of trying the simple functions increased from Task c) to d), but was still far below the students' results.

This comparison in the use of functions revealed that the students are more capable of building algorithms by using basic programming knowledge than the teachers. This

finding is in accordance with the high percentage of those teachers who were not able to decode the given formula and marked it as incorrect (Table 1). So the good news is that since the students are better at building algorithms than the teachers, there is hope for the future. The bad news is that the teachers' knowledge is not satisfactory.

Table 3: The name(s) of the functions remembered by the students and the teachers when solving Tasks c) and d)

	Task c)			Task d)		
	*IF?()	CSRA formula		*IF?()	CSRA formula	
		SUM()	IF()		AVERAGE()	IF()
2011/2012	13.6%	9.7%	14.2%	1.1%	9.4%	14.2%
2012/2013	15.4%	14.7%	17.0%	1.6%	10.8%	18.1%
teachers	52.2%	4.5%	4.5%	10.5%	6.72%	11.9%

Solving Task b) is different from Tasks a), c) and d) in at least two ways. First of all, in the formula of Task b) there is no function, only mathematical operators and references to cells. On the other hand this is the only task where the output is not a single result but a vector. Consequently, the way the original formula was multiplied should have been indicated in the solution. There are two possible ways to calculate the results for all the listed countries: one would be copying the formula created for the first country, the

other possibility is to use an array formula for the available countries. The first solution is more common and more widely used, while the second is the more flexible when a need emerges for modification.

Table 4: The evaluation of the in-execution-errors in Task b)

	division (/ operator)	order of the operands	population in thousands	all countries
2011/2012	44%	19%	9%	9%
2012/2013	44%	22%	8%	8%
teachers	69%	61%	35%	26%

The percentage of those selecting the right operator for division is approximately as high as those selecting the MAX() function in Task a). However, a very low number of students realized that the populations given in the table should be multiplied by 1000. Beyond that, only around 20% of the students knew that the population should be divided by the area, and decided on the right order of the operands. It was this which was the necessary background knowledge. Fewer than 10% realized that solving the problem will result in a vector.

The teachers' results are clearly higher than the students' but they also had a problem with the multiplication by 1000, and a more serious problem with creating a vector, both of which required knowledge gathered from the data in the table.

All of these errors can be classified as quantitative errors. However, while the selection of the wrong operator for the division, and the failure to create the vector are in-

execution-errors, swapping the numerator and the denominator, and not multiplying by 1000, are logical errors.

The follow-up test one year later

The test was repeated with the same students in their sophomore studies. Here students were divided into three groups. The groups were students who studied spreadsheets 1) with the CAAD-based approach, 2) with the "classical" TAEW-based approach, and 3) who did not study spreadsheets in their first year of studies. Group 4 were students who were not recognizable for various reasons; missing data or failed semesters.

Table 5: The students' results in the follow-up test in their sophomore year

	N	a)	b)	c)	d)	e)
Group 1	83	9.64%	1.20%	43.37%	21.69%	57.83%
Group 2	20	0.00%	0.00%	15.00%	0.00%	45.00%
Group 3	27	0.00%	0.00%	3.70%	0.00%	33.33%
Group 4	32	0.00%	0.00%	9.37%	0.00%	21.87%

The results for Task a) improved slightly in Group 1, but there were no correct solutions in the other groups. The reason for the better results in Group 1 is that they learned in their first year how to create multilevel formulas with the CAAD-based method. However, their results still need some improvement.

The result of Task b) did not change. They still did not know how to calculate the population density.
The results of Tasks c) and d) increased significantly in Group 1. The other two groups did not advance significantly.

Table 6: The use of the *IF?() functions and CSRA formulas when solving Tasks c) and d) in the follow-up test, in the students' sophomore year

	Task c)			Task d)		
	*IF?()	CSRA formula		*IF?()	CSRA formula	
		SUM()	IF()		AVERAGE()	IF()
Group 1	6.0%	73.5%	75.9%	0.0%	56.6%	56.6%
Group 2	10.0%	45.0%	65.0%	5.0%	50.0%	55.0%
Group 3	25.9%	22.2%	25.9%	3.7%	29.6%	48.1%
Group 4	0.0%	37.5%	50.0%	0.0%	37.5%	31.25%

When the functions and the sources of errors in the solutions to Tasks c) and d) were checked it was found that the students preferred to try solving the problems with the CAAD-based approach rather than the TAEW-based approach in all of the groups. The reason for this result would be that towards the end of their third semester the students had already completed more than three semesters of programming and algorithmic classes. It is still true that the percentage of the correct answers increased significantly only in Group 1, but the way the students approached the problems seems promising, although their

ability to give correct answers was hindered by a lack of proper instructions in spreadsheets.

Summary

It was found that the in-execution-errors of spreadsheet formulas are as frequent as any other types of errors. Students are familiar with simple, one-argument functions and the IF() function, but had problems using other two- or more-argument functions. All things considered, students prefer to learn and use simple functions rather than the complicated many-argument functions. If this is so, teachers have to focus on these functions and teach their students how to build multilevel formulas using simple functions and the IF() function.

It has also been proved with the automated error detections that longer formulas carry a lower number of errors than shorter ones (Panko and Aurigemma 2010). This finding and our results are in complete accordance. It can be explained by the fact that shorter formulas require a huge amount of special, rarely used background knowledge, whereas longer formulas are composed of simple functions.

The results of the follow-up tests proved that users can store simple functions in their long term memory and are able to build algorithms once they are familiar with the method.

Conclusions

The results of the previously known studies in error detection and our results in analyzing the sources of in-execution-errors lead us to recommend guidelines for

teachers of Informatics. The teachers' results show that they are in need of these guidelines.

What to teach?

Spreadsheets have to be taught from the CAAD-based point of view, focusing on the simple functions and on methods enabling the creation of multilevel functions and formulas based upon these simple functions. Functions which can be applied to various problems should be taught rather than those prepared expressly for solving special problems. The same is true in cases when more complicated, two- or three-argument functions are unavoidable. Beyond the selection of functions and multilevel formulas it is necessary to teach debugging, checking for the correctness of formulas and data, and the structure of the spreadsheet from the very beginning, as well as the right way to avoid errors in these documents.

How to teach?

The TAEW-based metacognitive approach, supported and publicized by most spreadsheet system developers, does not work in spreadsheet systems. Using the wizards does not develop conscious users. Users are not educated in computer sciences, so they do not understand the tags of the arguments in the wizards, preferring to just click here and there; they are happy when they are allowed to leave the wizard with an output value, without knowing whether it is correct or not. Consequently, the use of wizards should be omitted or only used rarely.

Instead of using the wizards, the text editor of the spreadsheet should be used for entering formulas, just like when creating source codes in high-level programming

languages. Beyond this, the phenomenon of functions introduced in Math classes should be imported into spreadsheet classes, and the other way around; the functions used in real world problems would strengthen the knowledge of functions. Consequently, both subjects and sciences would prosper with this twofold support.

How to evaluate the students' work?

In the process of evaluating the students' work changes are needed. Not only the final results but the in-execution-process should also be followed and evaluated. The process of building algorithms and the way students use functions and create multilevel functions and formulas should also be clearly detectable.

Notes: The research on which this article was based was supported by OTKA (K-105262).

References

- ABRAHAM, Robin and ERWIG, Martin (2009): Mutation Operators for Spreadsheets. In: Software Engineering, IEEE Transactions, 1.
- CASE Jennifer (2000): Students' perceptions of context, approaches to learning and metacognitive development in a second year chemical engineering course. [Unpublished PhD] Melbourne, Monash University.
- CASE Jennifer and GUNSTONE Richard (2002): Metacognitive development as a shift in approach to

learning: an in-depth study. In: Studies in Higher Education, 4, 459–470. p.
- CASE Jennifer and GUNSTONE Richard (2003): Approaches to learning in a second year chemical engineering course. In: International Journal of Science Education, 7, 801–819. p.
- CASE Jennifer, GUNSTONE Richard and LEWIS Alison (2001): Students' metacognitive development in an innovative second year chemical engineering course. In: Research in Science Education, 3, 331–355. p.
- COX Michael (2005): Metacognition in computation: A selected research review. In Artificial Intelligence, 2, 104–141. p.
- CSERNOCH Mária (2012): Introducing Conditional Array Formulas in Spreadsheet Classes. EDULEARN12 Proceedings. Barcelona, Spain. 2-4 July, 2012. Publisher: IATED, 7270–7279.
- CSERNOCH Mária and BALOGH László (2010): Algoritmusok és táblázatkezelés – Tehetséggondozás a közoktatásban az informatika területén. Magyar Tehetségsegítő Szervezetek Szövetsége, Budapest.
- CSERNOCH Mária and BIRÓ Piroska (2013a): Button-up technikák hatékonyságának vizsgálata informatika szakos hallgatók táblázatkezelés-oktatásában. Szerk: Kozma Tamás és Perjés istván, Új kutatások a neveléstudományokban, ELTE Eötvös Kiadó, 2013, Accepted.
- CSERNOCH Mária and BIRÓ Piroska (2013b): Teachers' Assessment and Students' Self-Assessment on The Students' Spreadsheet Knowledge. EDULEARN13 Proceedings July 1st-3rd, 2013 —

Barcelona, Spain. Publisher: IATED. ISBN: 978-84-616-3822-2. pp. 949–956.

- CSERNOCH Mária and BIRÓ Piroska (2013c): Spreadsheet misconceptions, spreadsheet errors. Hungarian Conference on Educational Research, Debrecen.
- CSÍKOS Csaba (2006): Metakogníció. A tudásra vonatkozó tudás pedagógiája. Műszaki Kiadó. Budapest.
- JORGENSEN Hugh (2013): How not to Excel in economics. In:http://www.lowyinterpreter.org/post/2013/04/18/How-not-to-Excel-in-economics.aspx (retrieved:17.06.2013.)
- KADIJEVICH Djordje (2009): Simple spreadsheet modeling by first-year business undergraduate students: Difficulties in the transition from real world problem statement to mathematical model. In BLOMHŘJ, M. and CARREIRA, S.(Eds.): Mathematical applications and modeling in the teaching and learning of mathematics: Proceedings the 11th International Congress on mathematical Education, Mexico, 241–248. p.
- KADIJEVICH Djordje (2013): Learning about spreadsheet. In KADIJEVICH Djordje, ANGELI Charoula and SCHULTE Carsten (Eds.) (2013): Improving Computer Science Education. New York/London, Routledge, 19–33. p.
- KORIAT Asher and LEVY-SADOT Ravit (2000): Conscious and Unconscious Metacognition: A Rejoinder. In: Consciousness and Cognition, 9, 193–202. p.

- KWAK James (2013): The Importance of Excel. http://baselinescenario.com/2013/02/09/the-importance-of-excel (retrieved: 17.05.2012)
- PANKO Raymond (2008): What We Know About Spreadsheet Errors. In: Journal of End User Computing's Special issue on Scaling Up End User Development, 2, 15–21. p.
- PANKO Raymond and AURIGEMMA Salvatore (2010): Revising the Panko-Halverson taxonomy of spreadsheet errors. In: Decision Support Systems, 2, 235–244. p.
- POWELL Stephen, BAKER Kenneth and LAWSON Barry (2008): A critical review of the literature on spreadsheet errors. In: Decision Support Systems, 1, 128–138. p.
- POWELL Stephen, BAKER Kenneth and LAWSON Barry (2009a): Errors in operational spreadsheets. In: Journal of Organizational and End-User Computing, 3, 4–36. p.
- POWELL Stephen, BAKER Kenneth and LAWSON Barry (2009b): Impact of errors in operational spreadsheets. Decision Support Systems, 2, 126–132. p.
- SCAFFIDI Christopher, SHAW Mary and MYERS Brad (2005): Estimating the Numbers of End Users and End User Programmers. In Proceedings of the 2005 IEEE Symposium on Visual Languages and Human-Centric Computing, 207–214. p.
- SESTOFT Peter (2010): Spreadsheet technology. Version 0.12 of 2012-01-31. IT University Technical Report ITU-TR-2011-142. Copenhagen, IT University of Copenhagen.

- TEO Thompson and TAN Margaret (1999): Spreadsheet Development and "What-if" Analysis: Quantitative versus Qualitative Errors. In: Accounting Management and Information Technologies, 9, 141–160.p.
- TORT Françoise (2010): Teaching Spreadsheets: Curriculum Design Principles. In THORNE, S. (Ed.), Proceedings of the EuSpRIG 2010 conference: Practical steps to protect organisations from out-of-control spreadsheets, p 99–110.
- TORT Françoise, BLONDEL François-Marie and BRUILLARD Éric (2008): Spreadsheet Knowledge and Skills of French Secondary School Students. R.T. Mittermeir and M.M. Sysło (Eds.): ISSEP 2008, LNCS 5090, Heidelberg, Springer-Verlag Berlin, 305–316. p.
- WALKENBACH John (2003): Excel 2003 Formulas. John Wiley & Sons.
- WALKENBACH John and WILCOX Colin (2003): Putting basic array formulas to work. In: http://office.microsoft.com/en-us/excel-help/putting-basic-array-formulas-to-work-HA001087292. aspx?CTT=5&origin=HA001087290. (retrieved: 08.05.2012)
- WILCOX Colin and WALKENBACH John (2003): Introducing array formulas in Excel: In: http://office.microsoft.com/en-us/excel-help/introducing-array-formulas-in-excel-HA001087290.aspx.(retrieved:18.01.2013.)
- W1 (2013): Meet the spreadsheet. In:http://office.microsoft.com/en-001/excel-help/meet-the-spreadsheet-

RZ101773335.aspx?section=2. (retrieved 02. 08. 2013.)

- W2 (2012): Report of JPMorgan Chase & Co. Management Task Force. Regarding (2012): CIO Losses. In: http://files.shareholder.com/downloads/ONE/2272984969x0x628656/4cb574a0-0bf5-4728-9582-625e4519b5ab/Task_Force_Report.pdf (retrived:17.05.2012.)
- W3 (2013): Database functions. In: http://office.microsoft.com/en-us/excel-help/excel-functions-by-category-HP010342656.aspx#BM database functions (07.06.2013.)
- W4 (2013): Lookup and reference functions. In: http://office.microsoft.com/en-us/excel-help/excel-functions-by-category-HP010342656.aspx#BM lookup_and_reference_functions (07.06.2013.)
- W5 (2012): Statistical functions. In: http://office.microsoft.com/en-us/excel-help/excel-functions-by-category-HP010342656.aspx#BM statistical functions. (retrieved: 08.05.2012)
- W6 (2013): Math and trigonometry functions. In: http://office.microsoft.com/en-us/excel-help/excel-functions-by-category-HP010342656.aspx#BM math and trigonometry functions. (retrieved: 07.06.2013.)
- W7 (2012): Az olasz, magyar, görög után most holland tanulmány a tudatlanság áráról. In:http://njszt.hu/ecdl/hir/20120312/az-olasz-magyar-gorog-utan-most-holland-tanulmany-a-tudatlansag-ararol(retrieved: 07.06.2013.)